GitHub Username: https://github.com/VassiliKurman

Route Tracker

Description

Route Tracker app tracks the routes of user by using on device GPS.

Intended User

This app is created for travelers, hikers and all individuals who want to keep record of their outdoor activities.

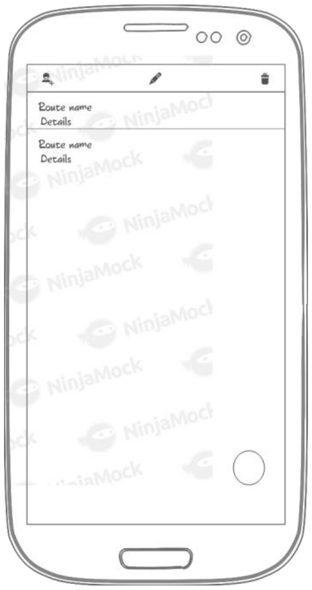
Features

Main features of the app:

- Saves route information into database
- Displays list of user routes previously saved into database
- Displays selected route on the Map
- Displays selected route summary on the screen
- User can share his/her routes with friends

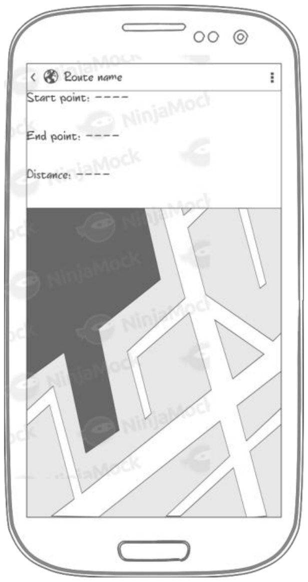
User Interface Mocks

Screen 1: Main activity with list of saved routes



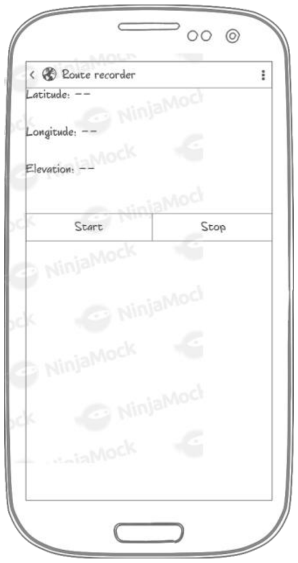
Main screen with list on saved routes and a floating action button to create new route.

Screen 2: Activity with route details



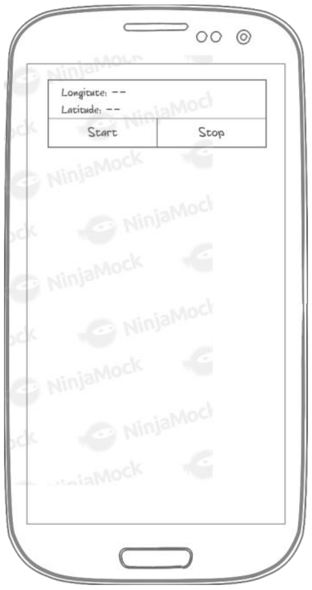
Screen to display selected route details with points on the map

Screen 3 Activity for new route tracking



Screen displaying Activity to create new route with current location details.

Screen 4: Widget



Screen with a widget to start and stop new route tracking from widget itself.

Key Considerations

How will your app handle data persistence?

- All data will be saved locally into device local database using Content Provider.

- Once new route is created, data also will be saved remotely on Firebase Realtime database.

Describe any edge or corner cases in the UX.

- App will retrieve data from local storage using Loaders and Content Provider
- App will send new route to Firebase Realtime Database using IntentService
- Back button natural behavior is not overridden.
- RTL layout switching will be enabled in manifest and layout
- Labeling UI elements, providing color contrast and grouping content will be used to support accessibility
- If there is no internet connection to display route details with map, only route summary will be displayed with notification about internet connection problem.
- If there is issue connecting to Firebase database, than notification will be displayed that data cannot be retrieved/saved from/to remote database.
- For any other error occurrence there will be Toast displayed stating the error cause.
- App will make use of resources saved in appropriate files, such as strings will be saved in strings.xml, common theme in styles.xml etc.
- Picasso will handle the image cashing and displaying
- Necessary icons will be saved in corresponding drawables folder

Describe any libraries you'll be using and share your reasoning for including them.

- App is written in Java language with JRE 1.8.0
- Android Studio version 3.1.3 is used as a development environment.
- Minimum SDK version 23 to find balance between features and number of devices that can use app
- Target SDK version 27
- Gradle version 4.4 is used as a build tool
- Picasso version 2.71828 to handle the loading and caching of images.
- Butter Knife version 8.8.1 for field and method binding.
- Google Play services SDK version 8.3 or later to use Map API
- Firebase Realtime Database version 16.0.1 for remotely saving/retrieving data and sharing capabilities

Describe how you will implement Google Play Services or other external services.

- Maps API to display user route on the Google Map.
- Firebase Realtime Database to keep all saved routes.

Next Steps: Required Tasks

Task 1: Project Setup

List of subtasks to setup the project in gradle:

- Configure Picasso library by adding dependency to gradle file
- Configure Butter Knife library by adding dependency to gradle file
- Configure Google Services to use Map API by adding necessary permissions and features to manifest
- Configure Firebase Realtime database by adding dependencies to gradle file

Task 2: Implement UI for Each Activity and Fragment

List the subtasks to create UI's for activities:

- Build UI for RoutesActivity with corresponding layout file to display list of saved routes and FAB to create new route
- Build UI for NewRouteActivity with corresponding layout file to start and stop new route recording
- Build UI for RouteDetailsActivity to display route summary
- Build MapFragment, which will be attached to RouteDetailsAcivity and will display route on the map
- Use Master/Detail flow to display list of routes and route details.

Task 3: Implement app logic

Implement app data creation, writing and reading logic:

- Load list of routes from local database and pass it to main activity
- If new route button is pressed, create new route object and save every data to local database while route is in progress using Content Provider
    - Once route is stopped, save route to Firebase Realtime Database **(to implement after task 4 completed)**
    - Share saved specific routes in Firebase database with friends **(to implement after task 4 completed)**
- If specific route is selected from the main activity, than display route details in new RouteDetailsActivity
    - Display routes on the Map **(to implement after task 5 is completed)**

Task 4: Implement Firebase services

Create database in Firebase Realtime Database:

- Setup Firebase database for this project
- Write data to database

- Read data from database

Task 5: Implement Google Play Services

Display map with selected route points:

- Add to route details activity fragment with Map
- Display route data as anchors/markers at specific positions on the map

Task 6: Implement Widget

Create new widget to start and stop new route tracking:

- Add widget to the project
- Implement logic to write data to ContentProvider once new route is started
- Once route tracking is stopped, display route summary in the widget.